Advanced Poirot with TensorFlow

Status: Draft
Author: haou, gTrade team
Last Updates: 2018-09-13
go/gtrade-advanced-poirot

Objective

Current Poirot/Marple model only use very limited features (exchange id, auction type, and bid bucket). In order to improve the Poirot surplus model quality, we plan to bring more features (e.g. query level features like domain URL, country, etc.) into the surplus model. The goal of this document is to propose a possible solution to this more complicated surplus model with query level features with TensorFlow.

Commented [1]: Thanks Hao for thinking about this.

Background Overview

Poirot (v1 design doc, v2 design doc)/Marple (design doc) project aims to protect advertisers' interests by lowering their first bids to maximize advertiser surplus when participating in non-second price auctions in external exchanges. The challenge here is that we are not aware of the auction dynamics in external exchanges. As a result, it's difficult to determine the optimal amount of bid lowering without losing too much impressions.

To resolve this issue, we use exploration experiments to figure out the auction dynamics in external exchanges by intentionally lowering the first bid to some predefined levels with bid multipliers. With the auction data collected from these exploration experiments, a quadratic model is trained between advertiser surplus and bid multipliers, and the optimal bid multipliers are chosen by maximizing the surplus using the quadratic model.

The advantages of the current Poirot/Marple model are

1. The current production model is small and simple, and it is easy to maintain and serve. Current production model only uses two features, exchange id and original first bid (in terms of bid bucket), plus auction type for Poirot model. With limited external exchanges we participate and number of bid buckets we choose, the number of feature combinations are relatively small, and the model can be easily served via dynamic files (which has size limitation that cannot support even mid-size models).
2. The current production models are fast to train. The training job for these models usually take only a few minutes, which means we can quickly update the production model on a new day.

These advantages must come with some drawbacks:

1. The model is small and simple, which means it may not have enough representation power to model the auction dynamics in external exchanges. As a result, we may get sub-optimal solutions.
2. The current production model is difficult to scale. If we want to include more query features to make the model more complicated and representative, the model serving file will quickly blow up.
3. Manual feature engineering is needed. We need to run experiments and do analysis to select good features to add into the model, which is time consuming and there is no guarantee that selected features are optimal.

Proposed Approach

With the data and computation resources available at Google scale, it seems that the advantages of the current production models are not that attractive if we could significantly improve the model performance by having a larger and more complicated Poirot/Marple surplus model. Thus, we propose to include more query level features into the Poirot/Marple surplus models, and implement them with TensorFlow.

TensorFlow Background

TensorFlow is not only for neural networks or deep learning. If you think neural networks as a function approximation to the underlying true function that generates the data, TensorFlow is just a tool to find the weights of the function approximator by solving the corresponding optimization problem. Thus, some traditional statistics model can also be trained with TensorFlow, e.g. linear regression, logistic regression, etc. Some core concepts in TensorFlow are tensor, operator, and computation graph. Generally speaking, tensors represents data, operators represents any operators conducted over tensors, e.g. add, matrix multiplication, etc., and computation graph represents the model we would like to build.

Proposed model

[redacted]

[redacted]

Infrastructure Support

If we could have some preliminary results to show that the proposed model have significant improvement over the current simple surplus model in production, we can migrate our prototype implementation to AdBrain infrastructure where they have comprehensive support for neural network models at scale within Ads, e.g. data collection, data storage (Woodshed), model training, monitoring, model validation, and model serving (via SeaStar, probably JellyFish in the near future).

TODO

1. Branch the current lingo pipeline to include more query features in the training data.
2. Build a prototype of the proposed surplus model and verify the prototype model performance offline.